

Towards a DSL for Perception-Based Safety Systems

Johann Thor Mogensen Ingibergsson* and Stefan-Daniel Suvei* and
Mikkel Kragh Hansen† and Peter Christiansen† and Ulrik Pagh Schultz*

* University of Southern Denmark, Campusvej 55, 5230 Odense M, Denmark

† Department of Engineering, Aarhus University, Finlandsgade 22, 8200 Aarhus N, Denmark

Email: {jomo|stdasu|ups}@mmmi.sdu.dk or {pech|mkha}@eng.au.dk

I. INTRODUCTION

Agriculture sees a high number of injuries and fatalities, even in developed countries [1]. Field robots are a solution to this problem, but the issue with field robots is that they are prone to failure [2], which has led to the use of Model-Driven Engineering (MDE). In robotics, MDE is used to improve development time and reliability, for example in regards to behavior [5]. In order to ensure safe autonomous operation, robust and reliable risk detection and obstacle avoidance must be performed. In this regard, field robots are highly dependent on perception sensors and algorithms to understand and react on the environment. The robot has to observe a large area; it must be fast, reliable and robust; it is constrained to function with low computational resources due to embedded hardware; and it might have lower priority than control and must not jam [3]. This imposes severe constraints on the software that interacts with sensors. Therefore, we propose a DSL [3] for safety concerns in perception systems. Our current research question is to investigate if the safety rules should be split between distinct domains (behavior [5] and perception [3]), or combined into one (this paper).

II. METHOD AND EXPERIMENTAL SETUP

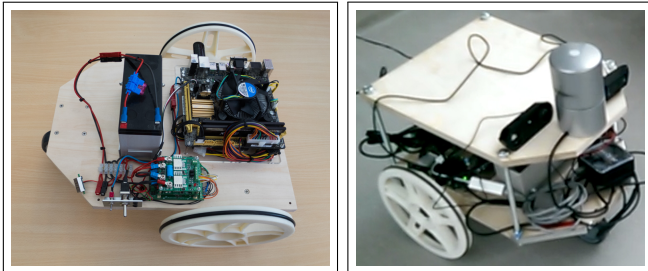


Fig. 1: Field robot prototype with lidar and camera sensors.

For investigating the research question, we built a field robot prototype (Fig. 1) to experiment with language design. First step is a test case implemented without MDE. To this end, we have extended a simple robot called Frobot, shown above, with non-trivial safety based on different sensors to prototype an agricultural field robot. The sensors are a Velodyne HDL-32E lidar with 32 channels for 360 degree 3D view and a Logitech C920 HD webcam. The prototype is designed for operation within an orchard. The lidar is used for detecting the orchard row to enable general navigation, and for object detection in front of the robot to enable obstacle avoidance [6]. The distances from the obstacles to the robot are used to adjust to the speed. The rows are found

using a Hough transform to enable the robot to position itself in the middle of the row. The camera detects humans using a C++ implementation of a pedestrian detector by Dollár [7]. A human position is estimated using the tilt of the camera, the bounding box position and a flat surface assumption. If it detects a human, the robot will sound warnings and limit the maximum speed according to the distance to the human, and ultimately bring the robot to a full stop.

Fig. 2 visualizes an example from an actual field trial. The red cylinder denotes a detected pedestrian, the blue lines denote the detected rows, and the yellow rectangle denotes the closest obstacle detected by the lidar. The traversable width of the row is decreased by the algorithm (indicated by the yellow line), such that the robot will effectively navigate around the closest obstacle. Due to a small distance to the detected pedestrian, the robot sounds a warning and decreases its speed.

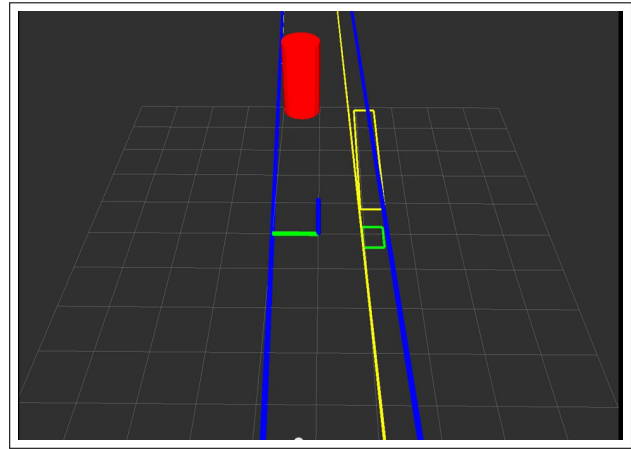


Fig. 2: Example of row navigation and obstacle avoidance. The robot has detected the rows (blue lines), a close obstacle (yellow rectangle), and a pedestrian (red cylinder). It adjusts its navigation (yellow line) and decreases its speed accordingly.

III. LANGUAGE DESIGN

We propose ideas of how to achieve sufficient safety levels for the perception system and the field robot as a whole. We want to create a DSL that combines behavioral safety [5] with our proposed perception safety system [3]. An outline of the DSL utilizing safety from both systems, can be seen below:

```

exists p in camera.all(pedestrian):
    distance(p) < 1m { sound emergency; stop; }
    distance(p) < 3m { sound move_away; cap_speed; }
    distance(p) < 5m { sound please_move_away; }
hist h = histogram(camera.image, bins = 10, normalized=
    true):
    size(x in h.bins: size(x)>0)/size(h.bins)<0.3 {
        cap_speed; }
    size(x in h.bins: size(x)>0)/size(h.bins)<0.1 { stop;
    }
    max(x in h.bins: size(x) > 0) - min(x in h.bins: size(
    x) > 0) < 1000px { stop; }
exists o in laser.all(Objects):
    distance(o) < 5m { cap_speed; }
    distance(o) < 0.5m { sound emergency; stop; }
lasers a in lasers(alive):
    count(a) < 32 { cap_speed; }
    count(a) < 26 { stop; }

```

Here, the output of the different sensors can be subjected to rules, that will enable the system to analyze the trustworthiness of the sensor and act accordingly. The first rule uses a pedestrian detector to calculate the distance to the nearest human and will then use cognitive safety to react, i.e., try to handle the situation before the emergency system kicks in. In addition to this, a functional layer performs a histogram analysis of the incoming images, verifying that the camera sensor is working. Similarly for a lidar, cognitive safety and functional safety are divided between different functionalities. In the first rule, the lidar detects objects using information from all available laser beams. Depending on the distance of an object, the speed of the robot is decreased accordingly. For the functional layer, it is possible to look at graceful degradation for the lidar, as it consists of many laser beams that provide a certain degree of redundancy. This is indicated by “*lasers(alive)*”. Functionality corresponding to the above code has been tested on the robot. The code *developed without the proposed language*, is 8454 lines of C++ and Python ROS code. The language that we propose could, if implemented, cut down the code to 14 lines. The next steps will be to follow MISRA [4] rules to extend the trustworthiness of the code into the domain of standards.

REFERENCES

- [1] Agricultural Statistics Board, “Agricultural Safety: 2009 Injuries to Adults on Farms,” 2013.
- [2] M. Reichardt, T. Föhst, and K. Berns, “On software quality-motivated design of a real-time framework for complex robot control systems,” in *Int. Workshop on Software Quality and Maintainability*, 2013.
- [3] J. T. M. Ingbergsson, U. P. Schultz, and D. Kraft, “Towards Declarative Safety Rules for Perception Specification Architectures,” submitted to *DSLRob Workshop 2015*, 2015.
- [4] MISRA, *MISRA-C Guidelines for the Use of the C Language in Critical Systems*. Motor Industry Software Reliability Assoc., 2012.
- [5] S. Adam, M. Larsen, K. Jensen, and U. P. Schultz, “Towards Rule-Based Dynamic Safety Monitoring for Mobile Robots,” in *SIMPAP*, Vol. 8810 in LNCS, pp. 207–218, Springer, 2014.
- [6] M. Kragh, R. N. Jørgensen, and H. Pedersen, “Object Detection and Terrain Classification in Agricultural Fields Using 3D Lidar Data,” in *10th International Conference on Computer Vision Systems, ICVS 2015*.
- [7] Piotr Dollár, Piotr’s Computer Vision Matlab Toolbox (PMT), at <http://vision.ucsd.edu/~pdollar/toolbox/doc/index.html>.